

Contents

1 Routine/Function Prologues	2
1.0.1 getdata.pl (Source File: getdata.pl)	2
1.0.2 ProcessArgs (Source File: getdata.pl)	3
1.0.3 SetEnv (Source File: getdata.pl)	4
1.0.4 BuildNames (Source File: getdata.pl)	5
1.0.5 CreateDir (Source File: getdata.pl)	6
1.0.6 BuildHoF (Source File: getdata.pl)	6
1.0.7 getsand (Source File: getdata.pl)	7
1.0.8 getclay (Source File: getdata.pl)	8
1.0.9 getalbedo (Source File: getdata.pl)	8
1.0.10 getcolor (Source File: getdata.pl)	9
1.0.11 getgfrac (Source File: getdata.pl)	9
1.0.12 getmask (Source File: getdata.pl)	10
1.0.13 getmaxsnalb (Source File: getdata.pl)	11
1.0.14 gettbot (Source File: getdata.pl)	11
1.0.15 getveg (Source File: getdata.pl)	12
1.0.16 getgeos (Source File: getdata.pl)	13

1 Routine/Function Prologues

1.0.1 getdata.pl (Source File: getdata.pl)

USES:

```
use warnings;
use strict;
```

INPUT PARAMETERS:

```
my $iam          # -- id of calling process
my $fullfilename # -- name of file to retrieve (including path)
my $slat         # -- southern latitude boundary of subset to retrieve
my $nlat         # -- norhtern latitude boundary of subset to retrieve
my $wlon         # -- western longitude boundary of subset to retrieve
my $elon         # -- eastern longitude boundary of subset to retrieve
my $qq           # -- quarter-season to retrieve (only for getalbedo)
my $mm           # -- month to retrieve (only for getgfrac)
my $index        # -- time-offset index (only for getgeos)
```

OUTPUT PARAMETERS:

Writes subset of data into fullfilename

DESCRIPTION:

This perl script is used to retrieve data files from LIS' GrADS-DODS data server

LOCAL VARIABLES:

```
my $filename;
my $filepath;
my $ctl;
my $ctlpath;

my $home_dir;
my $server;
my $log_dir;

my $data;
my %get;
```

CONTENTS:

```
&ProcessArgs(@ARGV);
&SetEnv;
&BuildNames;
&CreateDir;
&BuildHoF;
```

```

# Spread out gds requests
'sleep $iam';

# Remove the cache!
print " MSG: $0 -- Removing cache ( $iam )\n";
'rm -rf $home_dir/.dods_cache';

# Make GDS request
&{ $get{$data} };

print " MSG: $0 -- script done ( $iam )\n";

```

1.0.2 ProcessArgs (Source File: getdata.pl)

This subroutine processes the command line arguments to the main routine.

CONTENTS:

```

sub ProcessArgs
{
    my $called_as;
    my %data;

    %data = (
        "getsand"      => "SAND",
        "getclay"      => "CLAY",
        "getcolor"     => "COLOR",
        "gettbot"      => "TBOT",
        "getmask"      => "UMD Mask",
        "getveg"       => "UMD Veg",
        "getmaxsnalb"  => "MAXSNALB",
        "getalbedo"    => "ALBEDO",
        "getgfrac"     => "GFRAC",
        "getgeos"      => "GEOS",
    );

    ( $called_as = $0 ) =~ s::.*/(.*).pl:$1: ;
    $data = $data{$called_as} or die "ERR: $0 -- Called incorrectly";

    if ( $data eq "ALBEDO" )
    {
        ( $iam, $fullfilename, $slat, $nlat, $wlon, $elon, $qq ) = @_;
    }
    elsif ( $data eq "GFRAC" )

```

```

{
    ($iam, $fullfilename, $slat, $nlat, $wlon, $elon, $mm) = @_;
}
elsif ( $data eq "GEOS" )
{
    ($iam, $fullfilename, $index, $slat, $nlat) = @_;
}
else
{
    ($iam, $fullfilename, $slat, $nlat, $wlon, $elon) = @_;
}
}

```

1.0.3 SetEnv (Source File: *getdata.pl*)

This subroutine determines which system this script is being run on and sets several global variables.

CONTENTS:

```

sub SetEnv
{
    my $system;

    chomp( $system = `uname -s` );

    if ( $system eq "Linux" )
    {
        $server = "x6";
        $home_dir = "/home/jim";
        $log_dir = "/home/jim/$iam";
    }
    else
    {
        $server="lis1.sci.gsfc.nasa.gov";
        $home_dir=".";
        $log_dir ".";
    }
}

```

1.0.4 BuildNames (Source File: getdata.pl)

This subroutine takes the full file name of the data to be retrieved and breaks this name into the file path, file name, name of GrADS descriptor file, and path to the GrADS descriptor file.

CONTENTS:

```

sub BuildNames
{
    my $tmpctl;

    #filepath=${fullfilename%/*}
    #filename=${fullfilename##*/}
    #ctlpath=${filepath#/BCS/}
    #ctl=${filename%.*}

    if ( $data eq "SAND" or $data eq "CLAY" or $data eq "COLOR" or
        $data eq "MAXSNALB" or $data eq "TBOT" )
    {
        ($filepath = $fullfilename) =~ s:(.*)/.*:$1: ;
        ($filename = $fullfilename) =~ s:.*/: : ;
        ($ctlpath = $filepath)      =~ s:.*/BCS/: : ;
        ($ctl      = $filename)     =~ s:(.*)\..*:.$1: ;
    }
    elsif ( $data eq "GEOS" )
    {
        ($filepath = $fullfilename) =~ s:(.*)/.*:$1: ;
        ($filename = $fullfilename) =~ s:.*/: : ;
    }
    elsif ( $data eq "UMD Mask" or $data eq "UMD Veg" )
    {
        ($filepath = $fullfilename) =~ s:(.*)/.*:$1: ;
        ($filename = $fullfilename) =~ s:.*/: : ;
        ($ctlpath = $filepath)      =~ s:.*/GVEG/: : ;
        ($ctl      = $filename)     =~ s:(.*)\..*:.$1: ;
    }
    elsif ( $data eq "ALBEDO" or $data eq "GFRAC" )
    {
        ($filepath = $fullfilename) =~ s:(.*)/.*:$1: ;
        ($filename = $fullfilename) =~ s:.*/: : ;
        ($ctlpath = $filepath)      =~ s:.*/BCS/: : ;
        #($tmpctl = $filename)      =~ s:(.*)\..*:.$1: ;
        #($ctl   = $tmpctl)         =~ s:(.*)_.*_(.*):$1_$2: ;
        ($ctl      = $filename)     =~ s:(.*)_.*:$1: ;
    }

    print " DBG: $0 -- fullfilename $fullfilename ( $iam )\n";
    print " DBG: $0 -- filepath $filepath ( $iam )\n";
}

```

```

print " DBG: $0 -- filename $filename ( $iam )\n";
unless ( $data eq "GEOS" )
{
    print " DBG: $0 -- ctlpath $ctlpath ( $iam )\n";
    print " DBG: $0 -- ctl $ctl ( $iam )\n";
}
}

```

1.0.5 CreateDir (Source File: *getdata.pl*)

This subroutine creates the directory the write the retrieved data into.

CONTENTS:

```

sub CreateDir
{
    print " MSG: $0 -- Fetching $data data ( $iam )\n";

    if ( -e $filepath )
    {
        print " MSG: $0 -- $data directory $filepath exists ( $iam )\n";
    }
    else
    {
        print " MSG: $0 -- Creating $data directory $filepath ( $iam )\n";
        unless ( system("mkdir -p $filepath") )
        {
            print " MSG: $0 -- Created directory $filepath ( $iam )\n";
        }
        else
        {
            print " MSG: $0 -- Error occurred while creating directory $filepath ( $iam )\n";
            exit 1
        }
    }
}

```

1.0.6 BuildHoF (Source File: *getdata.pl*)

This subroutine creates a hash of functions to be used to call the appropriate data retrieval subroutine.

CONTENTS:

```

sub BuildHoF
{
    %get = (
        "SAND"      => \&getsand,
        "CLAY"      => \&getclay,
        "COLOR"     => \&getcolor,
        "ALBEDO"    => \&getalbedo,
        "GFRAC"     => \&getgfrac,
        "UMD Mask"  => \&getmask,
        "MAXSNALB"  => \&getmaxsnalb,
        "TBOT"       => \&gettbot,
        "UMD Veg"   => \&getveg,
        "GEOS"       => \&getgeos,
    );
}

```

1.0.7 getsand (Source File: getdata.pl)

This subroutine makes a system call to GrADS to retrieve the “sand fraction” soil data.
CONTENTS:

```

sub getsand
{
my $gradsstout;

$gradsstout =
`grads -bl <<EOB
sdfopen http://$server:dods/LIS_Params/${ctlpath}/${ctl}
set t 1
set x $wlon $elon
set y $slat $nlat
set fwrite -be -sq $fullfilename
set gxout fwrite
d sand
quit
EOB`;
#>> $log_dir/sand.$iam.log`

open(LOG, ">>$log_dir/sand.$iam.log");
print LOG $gradsstout;
close(LOG);
}

```

1.0.8 getclay (Source File: getdata.pl)

This subroutine makes a system call to GrADS to retrieve the “clay fraction” soil data.
CONTENTS:

```
sub getclay
{
my $gradsstdout;

$gradsstdout =
'grads -bl <<EOB
sdopen http://$server:9095/dods/LIS_Params/${ctlpath}/${ctl}
set t 1
set x $wlon $elon
set y $slat $nlat
set fwrite -be -sq $fullfilename
set gxout fwrite
d clay
quit
EOB';

open(LOG, ">> $log_dir/clay.$iam.log");
print LOG $gradsstdout;
close(LOG);
}
```

1.0.9 getalbedo (Source File: getdata.pl)

This subroutine makes a system call to GrADS to retrieve the albedo data.
CONTENTS:

```
sub getalbedo
{
my $gradsstdout;

$gradsstdout =
'grads -bl <<EOB
sdopen http://$server:9095/dods/LIS_Params/${ctlpath}/${ctl}
set t $qq
set x $wlon $elon
```

```

set y $slat $nlat
set fwrite -be -sq $fullfilename
set gxout fwrite
d alb
quit
EOB';

open(LOG, ">> $log_dir/alb.$iam.log");
print LOG $gradsstdout;
close(LOG);
}

```

1.0.10 getcolor (Source File: *getdata.pl*)

This subroutine makes a system call to GrADS to retrieve the soil color data.

CONTENTS:

```

sub getcolor
{
my $gradsstdout;

$gradsstdout =
'grads -bl <<EOB
sdopen http://$server:9095/dods/LIS_Params/${ctlpath}/${ctl}
set t 1
set x $wlon $elon
set y $slat $nlat
set fwrite -be -sq $fullfilename
set gxout fwrite
d soicol
quit
EOB';

open(LOG, ">>$log_dir/color.$iam.log");
print LOG $gradsstdout;
close(LOG);
}

```

1.0.11 getgfrac (Source File: *getdata.pl*)

This subroutine makes a system call to GrADS to retrieve the greenness fraction data.

CONTENTS:

```
sub getgfrac
{
my $gradsstdout;

$gradsstdout =
`grads -bl <<EOB
sdopen http://$server:9095/dods/LIS_Params/${ctlpath}/${ctl}
set t $mm
set x $wlon $elon
set y $slat $nlat
set fwrite -be -sq $fullfilename
set gxout fwrite
d gfrac
quit
EOB`;
open(LOG, ">>$log_dir/gfrac.$iam.log");
print LOG $gradsstdout;
close(LOG);
}
```

1.0.12 getmask (Source File: getdata.pl)

This subroutine makes a system call to GrADS to retrieve the UMD land/sea mask data.

CONTENTS:

```
sub getmask
{
my $gradsstdout;

$gradsstdout =
`grads -bl <<EOB
sdopen http://$server:9095/dods/LIS_Params/GVEG/${ctlpath}/${ctl}
set t 1
set x $wlon $elon
set y $slat $nlat
set fwrite -be -sq $fullfilename
set gxout fwrite
d tal
d nol
d mask
quit
```

```
EOB';
open(LOG, ">>$log_dir/mask.$iam.log");
print LOG $gradsstdout;
close(LOG);
}
```

1.0.13 getmaxsnalb (Source File: getdata.pl)

This subroutine makes a system call to GrADS to retrieve the maximum snow albedo data.
CONTENTS:

```
sub getmaxsnalb
{
my $gradsstdout;

$gradsstdout =
'grads -bl <<EOB
sdfopen http://$server:9095/dods/LIS_Params/${ctlpath}/${ctl}
set t 1
set x $wlon $elon
set y $slat $nlat
set fwrite -be -sq $fullfilename
set gxout fwrite
d snalb
quit
EOB';
open(LOG, ">>$log_dir/snalb.$iam.log");
print LOG $gradsstdout;
close(LOG);
}
```

1.0.14 gettbot (Source File: getdata.pl)

This subroutine makes a system call to GrADS to retrieve the “bottom temperature” data.
CONTENTS:

```
sub gettbot
{
my $gradsstdout;
```

```
$gradsstdout =
`grads -bl <<EOF
sdfopen http://$server:9095/dods/LIS_Params/${ctlpath}/${ctl}
set t 1
set x $wlon $elon
set y $slat $nlat
set fwrite -be -sq $fullfilename
set gxout fwrite
d tbot
quit
EOF`;
open(LOG, ">>$log_dir/tbot.$iam.log");
print LOG $gradsstdout;
close(LOG);
}
```

1.0.15 getveg (Source File: *getdata.pl*)

This subroutine makes a system call to GrADS to retrieve the UMD Vegetation classification data.

CONTENTS:

```
sub getveg
{
my $gradsstdout;

$gradsstdout =
`grads -bl <<EOF
sdfopen http://$server:9095/dods/LIS_Params/GVEG/${ctlpath}/${ctl}
set t 1
set x $wlon $elon
set y $slat $nlat
set fwrite -be -sq $fullfilename
set gxout fwrite
d tal
d nol
d veg1
d veg2
d veg3
d veg4
d veg5
d veg6
d veg7
d veg8
```

```

d veg9
d veg10
d veg11
d veg12
d veg13
quit
EOB';
open(LOG, ">>$log_dir/veg.$iam.log");
print LOG $gradsstdout;
close(LOG);
}

```

1.0.16 getgeos (Source File: getdata.pl)

This subroutine makes a system call to GrADS to retrieve the GEOS forcing data.
CONTENTS:

```

sub getgeos
{
my $gradsstdout;

$gradsstdout =
`grads -bl <<EOB
sdfopen http://$server:9095/dods/LIS_Forcing/GEOS/BEST_LK/geos3new
set t $index
set x 1 360
set y $slat $nlat
set fwrite -be -sq $fullfilename
set gxout fwrite
d t2m
d q2m
d radswg
d lwgdown
d u10m
d v10m
d ps
d preacc
d precon
d albedo
d sfctyp
d snow
d gwet
d t10m
d q10m

```

```
quit
EOB';
open(LOG, ">>$log_dir/geos.$iam.log");
print LOG $gradsstout;
close(LOG);
}
```